INFORMATION AND COMPUTATION 75, 232–263 (1987)

Safe States in Banker-like Resource Allocation Problems

DIRK HAUSCHILDT AND RÜDIGER VALK

Fachbereich Informatik, Universität Hamburg, Rothenbaumchaussee 67/69, D-2000 Hamburg 13, West Germany

This paper is concerned with methods of describing the set of safe states in the Banker's Problem. Using a Petri net model, formulas for this set (SAFE) and for its subset of minimal elements (MIN) are derived. Moreover, by partitioning MIN into subclasses such that elements of the same subclass differ only by a permutation of their components, an even smaller representation is given by a set SORT. Lower and upper bounds for the size of SORT are calculated. Since we give an algorithm which computes SORT in time linear to its size, these bounds are also applicable to the time complexity of computing SORT. Finally, some of the results are extended to the multidimensional Banker's Problem with different currencies, whereas other properties are shown to be not extendible to this case.

INTRODUCTION

The Banker's Problem was given by E. W. Dijkstra (1968) as an example of a resource sharing problem.

> A banker has n clients, and a fixed amount g of capital. Each client requires a predetermined amount, say f_i for the ith client, for his project. He does not need all the money at the beginning, but periodically he requests a unit of capital from the bank until his requirement is fulfilled. Some time later he returns his full loan to the bank. The banker may satisfy a given request if he has the money available, but he may choose not to. In that case the client has to wait until his request is satisfied. The Banker's Problem is to develop a strategy for distributing the money which will eventually satisfy all client's requirements. The banker has to avoid situations in which he has no money but there are clients' requests still outstanding. These situations are called deadlocks.

An instance of the problem is characterized by a positive integer n, an n-tuple $\mathbf{f}$, and a number g. All amounts are nonnegative integers. Given a particular problem instance, a state of the problem is an n-tuple $\mathbf{r}$ representing the amount required, but not yet received by each client. Initially, $\mathbf{r} = \mathbf{f}$. A state is safe if it does not necessarily lead to a deadlock.

0890-5401/87 $3.00

This paper is first concerned with finding the minimal safe states. (n-tuples are partially ordered in the usual way, and once a state is known to be safe, an increase in any component does not lead to an unsafe state.) In fact, it is shown that all minimal safe states can be obtained by first sorting the f_i's in ascending order, and then computing the set SORT (the minimal safe states which have sorted r_i's). All minimal safe states are permutations of an element of SORT. Since one element of SORT thus represents several actual minimal safe states, this is an efficient method of describing the safe states.

We present two algorithms. The first one tests if a given state is safe when sorted permutations of the f_i's and r_i's are given. The other one computes the set SORT from the f_i's. The second concern of the paper is in finding upper and lower bounds on the size of the set SORT. The calculation makes essential use of the classical mathematics of partitions.

In the final section an extension of the problem is discussed where different types of resources have to be controlled. Since such resources can be interpreted as different unconvertible currencies we call this extension the "international" Banker's Problem. It is shown to what extent the results of this paper also apply to this case.

We use Petri nets as a suitable operational model for the banker's actions. The reachable states can be computed by the help of invariants in the net and coloured Petri nets give us the opportunity to extend our model to the international case without drastically increasing the size of the net.

Additionally, the monotonicity mentioned above (safety is preserved when increasing r) corresponds to the so-called monotonicity of firings in Petri nets. (I.e., a sequence of transitions is firable at all markings $\mathbf{m}' > \mathbf{m}$ if it is firable at $\mathbf{m}$.) Thus properties of markings like unboundedness or T-continuality (for a definition see below) are conserved when increasing a marking. In Valk and Jantzen (1985) it was shown that for a set of markings with such a property its (finite) set of minimal markings, building a description of the whole set, is effectively computable. We will identify the safe states in the Banker's Problem with the T-continual markings in our net description of the problem. Thus Section 2 of the present work can be seen as an application of the methods introduced in Valk and Jantzen (1985).

We would like to thank the anonymous referees for their thorough examination of this text. Their remarks helped us much to improve the representation of this paper and to eliminate errors.

1. Basic Definitions

A **place/transition net** (P/T-net) $PTN = (P, T, F, B, \mathbf{m}_0)$ is defined by a finite set P of **places**, a finite set T of **transitions**, disjoint from P, two mappings:

$$F: P \times T \to \mathbb{N},$$

$$B: P \times T \to \mathbb{N},$$

and an initial marking $\mathbf{m}_0$.

F and B are called the **forward** and **backward indicence mapping**. They can also be seen as $(|P| \times |T|)$-matrices over $\mathbb{N} = \{0, 1, ...\}$. Let $\Delta := B - F$ be the incidence matrix of PTN. (The symbol := will mean "is defined as" in this paper.) $F(t)$ ($B(t)$, $\Delta(t)$) denotes the t-column of $F(B, \Delta)$. A marking $\mathbf{m}$ is a function from P into $\mathbb{N}$ and can be seen as vector of length $|P|$.

Usually, a net is defined by giving a graphical representation of it rather than mentioning the quintupel explicitly. Places are drawn as circles while transitions are represented by rectangles. There is an arc from place p to transition t labelled $F(p, t)$ unless this value is zero. Labels of one are omitted. Using the same method, B is depicted by arcs from transitions to places. The initial marking of a place is written into the circle representing it. Again, zeroes are omitted.

A transition $t \in T$ has **concession** in $\mathbf{m}$, written $\mathbf{m}(t\rangle$, if $\mathbf{m} \geqslant F(t)$. t converts $\mathbf{m}$ into $\mathbf{m}'$, written $\mathbf{m}(t\rangle \mathbf{m}'$, if $\mathrm{m}(t\rangle$ and $\mathbf{m}' = \mathbf{m} + \Delta(t)$. This notion is extended to finite words $w \in T^*$ and infinite words $w \in T^\omega$ in the usual manner.

A marking $\mathbf{m}$ is called **T-continual** if there is an infinite sequence $w \in T^\omega$ that contains every transition from T infinitely often and has concession in $\mathbf{m}$.

For any word $w \in T^* \cup T^w$ let w_ν be the νth transition in w and $[w]_\nu = \prod_{i=1}^{\nu} w_i$ the prefix of length ν. We identify functions of type f: $\{1, ..., n\} \to \mathbb{N}$ with the corresponding vectors of length n. $\mathrm{PERM}\{1, ..., n\}$ denotes the set of bijections on $\{1, ..., n\}$. We extend this notion to functions f from $\{1, ..., n\}$ to $\mathbb{N}$ by PERM $f :=$ $\{f \circ p \mid p \in \mathrm{PERM}\{1, ..., n\}\}$. $f \circ p(i)$ means $f(p(i))$. In this paper we sometimes define permutations by sorting the values of a function: $p \in \mathrm{PERM}\{1, ..., n\}$ is **f-sorted**, if and only if

$$\underset{i,j \in N}{\forall} \; i \leqslant j \Rightarrow f(p(i)) \leqslant f(p(j)).$$

For example, given f by $(f(1), f(2), f(3)) = (8, 3, 9)$, then $(2, 1, 3)$ is the only f-sorted permutation of $\{1, 2, 3\}$.

2. Safe States and Minimal Elements

The P/T-net in Fig. 2.1 represents the Banker's Problem as described in the Introduction. The place BANK, holding the banker's cash, initially contains g units of money. CREDIT$_i$ and CLAIM$_i$ stands for the loan and the remaining claim of the client i, respectively. By the transition GRANT$_i$ this client obtains one unit of money as often as GRANT$_i$ fires. RETURN$_i$ returns all the money back to the banker. RETURN$_i$ cannot fire before the banker has fulfilled the maximal claim f_i of the client. By the same transition this claim is restored in CLAIM$_i$.

An arbitrary marking in the net is a vector $(r_1, ..., r_n, k_1, ..., k_n, b)$, where r_i, k_i and b are the numbers of tokens in the places CLAIM$_i$, CREDIT$_i$, and BANK, respectively. Hence the initial marking is $\mathbf{m}_0 = (f_1, ..., f_n, 0, ..., 0, g)$.

The following invariant equations hold for every reachable marking:

$$I_0. \quad \sum_{i=1}^{n} k_i + b = g;$$

$$I_i. \quad k_i + r_i = f_i \; (1 \leqslant i \leqslant n).$$

By these invariants equations every marking is uniquely described by giving $(r_1, ..., r_n)$. Therefore we will use this simplified version and denote the whole marking by using the map

$$m(\mathbf{r}) := (r_1, ..., r_n, k_1, ..., k_n, b)$$

with

$$k_i := f_i - r_i \qquad \text{and} \qquad b := g - \sum_{i=1}^{n} (f_i - r_i).$$

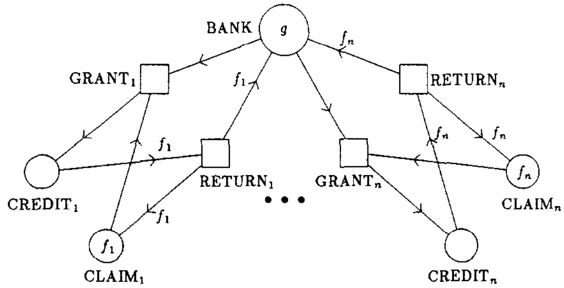

FIG. 2.1. The Banker's net.

We extend this mapping to sets of vectors R by $m(R) := \{m(\mathbf{r}) \mid \mathbf{r} \in R\}$. $\mathbf{f} = (f_1, ..., f_n)$ describes the initial marking, i.e., $\mathbf{m}_0 = m(\mathbf{f})$.

Conversely, these invariant equations are also a sufficient reachability criterion as we will show in the next lemma. Hence, they fully describe the set of reachable markings. Using these equations, we then are able to characterize the reachable vectors by defining a set REACH. This will be justified by Theorem 2.2.

LEMMA 2.1. *Every vector from $\mathbb{N}^{2n+1}$ satisfying the invariant equations is a reachable marking.*

Proof. A vector $\mathbf{m} = (r_1, ..., r_n, k_1, ..., k_n, b)$ satisfying I_0 and I_i $(1 \leqslant i \leqslant n)$ is reachable from $\mathbf{m}_0$ by $w = \prod_{i=1}^{n} \mathrm{GRANT}_i^{k_i}$, i.e., $\mathbf{m}_0(w\rangle\, \mathbf{m}$. ∎

We describe the reachable markings by the set

$$\mathrm{REACH} := \left\{ \mathbf{r} \in \mathbb{N}^n \,\middle|\, \text{(a)} \ \forall_{1 \leqslant i \leqslant n} \ r_i < f_i, \qquad \text{(b)} \ \sum_{i=1}^{n} r_i \geqslant \sum_{i=1}^{n} f_i - g \right\}.$$

THEOREM 2.2. *$m(\mathrm{REACH})$ is the set of reachable markings.*

Proof. Let $\mathbf{m} = (r_1, ..., r_n, k_1, ..., k_n, b)$ be an arbitrary element of $m(\mathrm{REACH})$. By definition we have $r_i \geqslant 0$ and from $r_i \leqslant f_i$ follows $k_i \geqslant 0$ for all $1 \leqslant i \leqslant n$. From $b = g - \sum_{i=1}^{n} (f_i - r_i) \geqslant 0$ follows that $\mathbf{m} \in \mathbb{N}^{2n+1}$. Therefore $\mathbf{m}$ is a reachable marking by Lemma 2.1. Conversely if $\mathbf{m} \in \mathbb{N}^{2n+1}$ is reachable, the invariant equations hold for $\mathbf{m}$, i.e., $\mathbf{m} \in m(\mathrm{REACH})$. ∎

Starting with the two conditions (a) and (b) in the definition of REACH we now successively add other conditions to describe the set SAFE. The first of them is named (c). In order to obtain alternative and more effective characterisations of SAFE we then refine condition (c) to (c′) and (c″). Thus we get three equivalent set definitions: SAFE, SAFE′, and SAFE″. For each of these versions we define the set MIN (MIN′, MIN″) of its minimal elements as well. This will be done by replacing (b), (c), (c′), and (c″) by ($\mathrm{b}_{\min}$), ($\mathrm{c}_{\min}$), ($\mathrm{c}'_{\min}$), and ($\mathrm{c}''_{\min}$), respectively. Fig. 2.2 gives an overview of the sets and conditions as defined so far.

$$\begin{array}{ccccc}
\underset{\{\mathbf{r}\in\mathbb{N}^n \mid (\mathrm{a}) \text{ and } (\mathrm{b})\}}{\mathrm{REACH}} & \supseteq & \underset{\{\mathbf{r}\in\mathbb{N}^n \mid (\mathrm{a}), (\mathrm{b}), \text{ and } (\mathrm{c})\}}{\mathrm{SAFE}} & \supseteq & \underset{\{\mathbf{r}\in\mathbb{N}^n \mid (\mathrm{a}), (\mathrm{b}_{\min}) \text{ and } (\mathrm{c}_{\min})\}}{\mathrm{MIN}} \\
 & & = & & = \\
 & & \underset{\{\mathbf{r}\in\mathbb{N}^n \mid (\mathrm{a}), (\mathrm{b}), \text{ and } (\mathrm{c}')\}}{\mathrm{SAFE}'} & \supseteq & \underset{\{\mathbf{r}\in\mathbb{N}^n \mid (\mathrm{a}), (\mathrm{b}_{\min}) \text{ and } (\mathrm{c}'_{\min})\}}{\mathrm{MIN}'} \\
 & & = & & = \\
 & & \underset{\{\mathbf{r}\in\mathbb{N}^n \mid (\mathrm{a}), (\mathrm{b}) \text{ and } (\mathrm{c}'')\}}{\mathrm{SAFE}''} & \supseteq & \underset{\{\mathbf{r}\in\mathbb{N}^n \mid (\mathrm{a}), (\mathrm{b}_{\min}) \text{ and } (\mathrm{c}''_{\min})\}}{\mathrm{MIN}''}
\end{array}$$

FIG. 2.2. The structure of the defined sets and their conditions.

In the last chapter we will have a look at the "international" Banker's Problem. The corresponding definitions are distinguished from the national versions by a prefix i, i.e., ISAFE, IMIN, (ia), (ib), and so on.

A state is safe if and only if the claims of all clients can be satisfied in some sequence. This means that there is a permutation $p \in \mathrm{PERM}\{1, ..., n\}$ such that in step i the remaining claim $r_{p(i)}$ of client $p(i)$ does not exceed the banker's actual cash b plus the amount $\sum_{j=1}^{i-1} k_{p(j)}$ of money he got back in the previous steps. (If the range of summation is empty, the result is 0.) Hence, safety is equivalent to

$$\exists p \in \mathrm{PERM}\{1, ..., n\} \quad \forall_{1 \leqslant i \leqslant n} \quad r_{p(i)} \leqslant b + \sum_{j=1}^{i-1} k_{p(j)}.$$

This is equivalent to condition (c) in the following definition of SAFE, where $k_{p(i)}$ is replaced by $f_{p(i)} - r_{p(i)}$:

$$\mathrm{SAFE} := \left\{ \mathbf{r} \in \mathbb{N}^n \mid \begin{array}{ll} \text{(a)} & \forall_{1 \leqslant i \leqslant n} \; r_i \leqslant f_i, \\ \text{(b)} & \sum_{i=1}^{r} r_i \geqslant \sum_{i=1}^{n} f_i - g, \\ \text{(c)} & \exists p \in \mathrm{PERM}\{1, ..., n\} \\ & \forall_{1 \leqslant i \leqslant n} \sum_{j=1}^{i} r_{p(j)} \leqslant b + \sum_{j=1}^{i-1} f_{p(j)}, \\ & b \text{ can be expressed in terms of } t \text{ and the} \\ & \text{constant values } f_i \text{ and } g \text{ in the following way:} \\ & b = g - \sum_{i=1}^{n} f_i + \sum_{i=1}^{n} r_i \end{array} \right\}.$$

We first consider the question whether the initial state $m(\mathbf{f})$ is safe.

LEMMA 2.3. *The following conditions are equivalent*:

(i) $\forall_{1 \leqslant i \leqslant n} f_i \leqslant g$.

(ii) $\mathbf{f} \in \mathrm{SAFE}$.

(iii) $\mathrm{SAFE} \neq \varnothing$.

(iv) $m(\mathbf{f})$ *is T-continual.*

(v) *There is a T-continual, reachable marking in the net.*

Proof. i ⇒ ii. Setting $\mathbf{r} = \mathbf{f}$ makes (a) and (b) true; (c) is transformed into i.

ii ⇒ iii. This is obvious.

iii $\Rightarrow$ i. Assume $\mathbf{r} \in \text{SAFE}$. Then there is a permutation p satisfying for every i:

$$\sum_{j=1}^{i} r_{p(j)} \leqslant b + \sum_{j=1}^{i-1} f_{p(j)}.$$

Thus

$$\begin{aligned} r_{p(i)} &\leqslant b + \sum_{j=1}^{i-1} f_{p(j)} - \sum_{j=1}^{i-1} r_{p(j)} \\ &= b + \sum_{j=1}^{i-1} k_{p(j)} \\ &\leqslant g - k_{p(i)} = g - f_{p(i)} + r_{p(i)} \end{aligned}$$

and $f_{p(i)} \leqslant g$ holds as well.

i $\Leftrightarrow$ iv. Since $(\prod_{i=1}^{n} w_i)^{\omega}$ with $w_i = \text{GRANT}_i^{f_i}\,\text{RETURN}_i$ has concession in $\mathbf{m}_0$ if $f_i \leqslant g$ for all $1 \leqslant i \leqslant n$. On the other hand, if $f_j > g$, then RETURN_j never has concession.

iv $\Leftrightarrow$ v. v follows from iv directly. In order to prove that v implies iv observe that $\mathbf{m}(w\rangle$ and $\mathbf{m}_0(v\rangle\,\mathbf{m}$ implies $\mathbf{m}_0(vw\rangle$. ∎

THEOREM 2.4. *$m(\text{SAFE})$ is the set of reachable and T-continual markings.*

Proof. Assume $\mathbf{r} \in \text{SAFE}$. By Lemma 2.1 $\mathbf{m} := m(\mathbf{r})$ is reachable from $\mathbf{m}_0$. In order to show that $\mathbf{m}$ is T-continual, it suffices to prove that $\mathbf{m}_0$ is reachable from $\mathbf{m}$, since $\mathbf{m}_0$ is T-continual from Lemma 2.3.

Let p be a permutation satisfying condition (c) in the definition of SAFE. Then we define $w_i := \text{GRANT}_i^{r_i}\,\text{RETURN}_i$ and $w := \prod_{i=1}^{n} w_{p(i)}$ for $\mathbf{r} = (r_1, ..., r_n)$. Using this condition it is not difficult to prove $\mathbf{m}(w\rangle\,\mathbf{m}_0$. Intuitively, w describes a way that the banker can regain all his money.

Conversely, let m($\mathbf{r}$) be a reachable and T-continual marking. We have to show that conditions (a), (b), and (c) in the definition of SAFE are satisfied. (a) and (b) follow from Theorem 2.2. Therefore, condition (c) remains to be proved.

Since m($\mathbf{r}$) is T-continual, there is some $w \in T^{\omega}$ having concession in $m(\mathbf{r})$ and containing all $t \in T$. Let v_i $(1 \leqslant i \leqslant n)$ denote the position of the first occurrence of RETURN_i in w, and p the v-sorted permutation of $\{1, ..., n\}$. Then $\text{RETURN}_{p(j)} \notin [w]_{v_i}$ for $j > 1$. Using p, we now prove for all $1 \leqslant i \leqslant n$

$$\sum_{j=1}^{i} r_{p(j)} \leqslant b + \sum_{j=1}^{i-1} f_{p(j)}.$$

Suppose that the inequality does not hold for some i. We define $w' := [w]_{v_{p(i)}-1}$ and S as the sum of all tokens in the places BANK and $\text{CREDIT}_{p(j)}$ for $1 \leqslant j \leqslant i$. Then

$$S = b + \sum_{j=1}^{i} k_{p(j)} = b + \sum_{j=1}^{i} f_{p(j)} - \sum_{j=1}^{i} r_{p(j)} < f_{p(i)}.$$

S can be increased only by the firing of some transitions $\text{RETURN}_{p(j)}$ with $j > i$, which do not appear in w'. Hence, after firing of w', the number of tokens in $\text{CREDIT}_{p(i)}$ is maximally $S < f_{p(i)}$. Therefore $\text{RETURN}_{p(i)}$ has no concession, contradicting the definition of w.

As long as the banker has not recovered any money from clients $p(i+1)$ to $p(n)$, he cannot pay the whole credit to client $p(i)$. Thus w does not describe a way of getting the client's loan back. ∎

The set of T-continual markings of every P/T-net is right closed (see Valk and Jantzen, 1985), i.e., if $\mathbf{m}$ is T-continual and $\mathbf{m}' \geqslant \mathbf{m}$, then also $\mathbf{m}'$ is T-continual. This allows us to describe this set by its subset of minimal elements, which is finite.

In this paper we are restricted to the (finite) set of markings reachable from a fixed initial marking $\mathbf{m}_0$. Here a similar property is valid. If $\mathbf{r} \in \text{SAFE}$ and $\mathbf{r}' \geqslant \mathbf{r}$ is reachable, then also $\mathbf{r}' \in \text{SAFE}$. This can be proved by using condition (c) in the definition of SAFE, where b is eliminated: We get

$$\sum_{j=1}^{i} r_{p(j)} \leqslant g - \sum_{j=1}^{n} f_j + \sum_{j=1}^{n} r_j + \sum_{j=1}^{i-1} f_{p(j)}$$

and hence

$$\sum_{j=i}^{n} f_{p(j)} \leqslant g + \sum_{j=i+1}^{n} r_{p(j)}.$$

The last inequality remains valid when substituting $\mathbf{r}$ by $\mathbf{r}' \geqslant \mathbf{r}$, since f_i and g are constant. Therefore, we can describe the safe states also by the smaller set MIN of SAFE's minimal elements:

$$\text{MIN} := \Big\{ \mathbf{r} \in \mathbb{N}^n \mid \text{(a)} \quad \forall_{1 \leqslant i \leqslant n}\ r_i \leqslant f_i,$$
$$(\text{b}_{\min}) \quad \sum_{i=1}^{n} r_i = \max\left(0, \sum_{i=1}^{n} f_i - g\right),$$
$$(\text{c}_{\min}) \quad \exists p \in \text{PERM}\{1, ..., n\}$$
$$\forall_{1 \leqslant i \leqslant n}\ \sum_{j=1}^{i} r_{p(j)} \leqslant \sum_{j=1}^{i-1} f_{p(j)} \Big\}.$$

THEOREM 2.5. *MIN is the set of minimal elements of SAFE.*

Proof. Before showing the inclusions we observe that, if (a) and (b_{min}) hold, (c) is equivalent to (c_{min}):

(a) Every $\mathbf{r} \in$ MIN is a minimal element of SAFE: By (b_{min}) $\Rightarrow$ (b) we have $\mathbf{r} \in$ SAFE. If $\mathbf{r}' \leqslant \mathbf{r}$ for some $\mathbf{r}'$, then $\sum_{i=1}^{n} r'_i < \sum_{i=1}^{n} f_i - g$. Hence $\mathbf{r}' \notin$ SAFE.

(b) Every minimal element of SAFE belongs to MIN: Let $\mathbf{r} \in$ SAFE$\backslash$MIN, i.e.,

$$\sum_{i=1}^{n} r_i > \max\left(0, \sum_{i=1}^{n} f_i - g\right).$$

Let p be a permutation of $\{1, ..., n\}$ satisfying (c) and j the smaller index with $r_{p(j)} > 0$. Then we will show that

$$\mathbf{r}' = (r_1, ..., r_{p(j)-1}, r_{p(j)} - 1, r_{p(j)+1}, ..., r_n) \in \text{SAFE}.$$

and thus $\mathbf{r}$ is not minimal in SAFE.

$\mathbf{r}'$ clearly satisfies (a) and (b). We prove that $\mathbf{r}'$ satisfies (c) when using the same permutation p. This holds for $i' \geqslant j$, since both sides of the inequality in (c) are decreased by one. For $i' > j$, the left-hand side vanishes. By $\sum_{i=1}^{n} r'_i = \sum_{i=1}^{n} r_i - 1$ we have

$$b = g - \sum_{i=1}^{n} f_i + \sum_{i=1}^{n} r'_i = g - \sum_{i=1}^{n} f_i + \sum_{i=1}^{n} r_i - 1 \geqslant 0.$$

Therefore the whole right-hand side of (c) is at least zero. ∎

We conclude this Section by giving an example describing what the sets REACH, SAFE, and MIN look like. Figure 2.3 depicts these sets for the instance of the Banker's Problem discussed in Brinch Hansen (1973): $n = 3$ clients have claims $f_1 = 8$, $f_2 = 3$, and $f_3 = 9$. The banker's capital is $g = 10$. The resulting P/T-net in the form of Fig. 2.1 has 197 reachable markings. They are represented as circles in the picture. 24 of these markings are deadlocks. The 137 T-continual markings are depicted by the white circles, which contain a cross if they are minimal. For each of the ten minimal elements $\mathbf{min}_i$, we define the set

$$M_i := \{\mathbf{r} \in \text{REACH} \mid \mathbf{r} \geqslant \mathbf{min}_i\}.$$

Just like MIN generates SAFE $= \bigcup_{i=1}^{10} M_i$ we can generate subsets of SAFE by subsets of MIN. If MIN is very large, a control mechanism can use a small subset of MIN to generate a big subset of SAFE. Then the

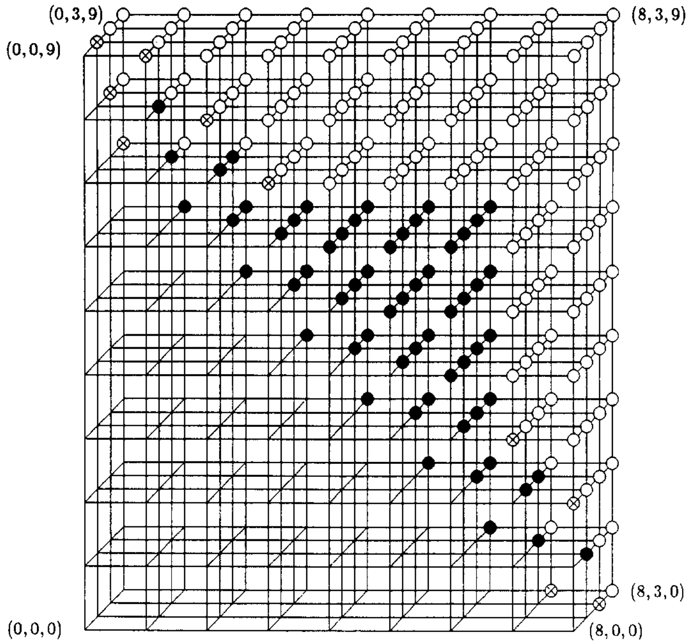

FIG. 2.3. REACH and SAFE for an instance of the Banker's Problem.

algorithm will exclude some, but not all, possible transactions. Figure 2.3, for example, leads to the idea to cut off the big black corner in the middle. Figure 2.4 shows that maximal possible number of represented T-continual

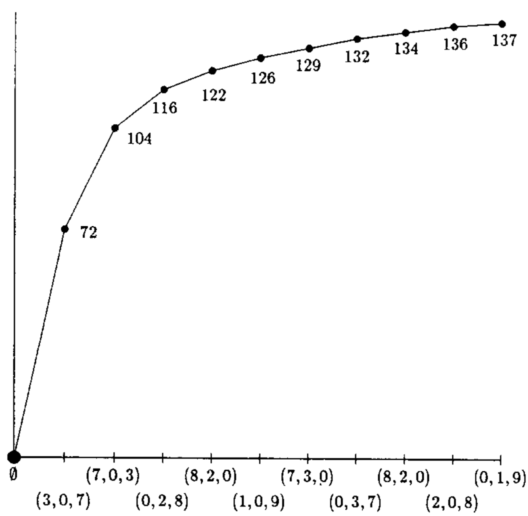

FIG. 2.4. Covering subsets of SAFE.

markings as a function of the number of elements of S. For instance, by choosing two particular elements of MIN, namely $(3, 0, 7)$ and $(7, 0, 3)$, 104 elements of SAFE are covered.

3. Efficient Use of Permutations

The definitions of SAFE and MIN contain existential quantifiers ranging over all permutations of n (= number of clients) elements. In this chapter we will show that this computational complexity is not inherent to the problem.

To be more precise, for every element $\mathbf{r} \in \text{SAFE}$, every reachable permutation $\mathbf{r}'$ of $\mathbf{r}$ belongs to SAFE as well. Hence, when we define the equivalence relation $\mathbf{r} \equiv \mathbf{r}' \Leftrightarrow \mathbf{r}' \in \text{PERM}\, \mathbf{r}$ on SAFE and MIN, we need only one representative of each equivalence class to entirely describe these sets.

First we give alternative definitions, called SAFE$'$ and MIN$'$, for SAFE and MIN, respectively. Then we prove them to be equivalent to the former ones.

$$\text{SAFE}' := \Big\{ \mathbf{r} \in \mathbb{N}^n \mid \text{(a)} \quad \forall_{1 \leqslant i \leqslant n} \; r_i \leqslant f_i,$$

$$\text{(b)} \quad \sum_{i=1}^{n} r_i \geqslant \sum_{i=1}^{n} f_i - g,$$

$$\text{(c}'\text{)} \quad \exists p \in \text{PERM}\{1, ..., n\} \; \forall q \in \text{PERM}\{1, ..., n\}$$

$$\forall_{1 \leqslant i \leqslant n} \sum_{j=1}^{i} r_{p(j)} \leqslant b + \sum_{j=1}^{i-1} f_{q(j)}$$

$$\text{with } b = g - \sum_{i=1}^{n} f_i + \sum_{i=1}^{n} r_i \Big\}.$$

For proving SAFE $=$ SAFE$'$ it is sufficient to show (c) $\Rightarrow$ (c$'$), since (c$'$) $\Rightarrow$ (c) is obvious. This proof is divided into the following four lemmas.

Lemma 3.1. *Let $m(\mathbf{r})$ be a reachable marking and p be a permutation of $\{1, ..., n\}$ satisfying*

$$\forall_{1 \leqslant i \leqslant n} \sum_{j=1}^{i} r_{p(j)} \leqslant b + \sum_{j=1}^{i-1} f_{p(j)}.$$

If q is an **r***-sorted permutation of* $\{1, ..., n\}$, *then*

$$\underset{1 \leqslant i \leqslant n}{\forall} \sum_{j=1}^{i} r_{q(j)} \leqslant b + \sum_{j=1}^{i-1} f_{q(j)}.$$

Proof. The lemma has to be proved for an arbitrary index i satisfying $1 \leqslant i \leqslant n$. Let i' be the smallest index with $r_{p(i')} \geqslant r_{q(i)}$. P is defined to be the set $\{1, ..., i'-1\}$ Hence for all elements j of $Q := q^{-1}(p(P))$ we obtain $r_{q(j)} < r_{q(i)}$. Since q is r-sorted, j must be smaller than i.

We now have a situation as in Fig. 3.1, where $(r_{q(1)}, ..., r_{q(n)})$ is piled up on the left-hand side and $(f_{p(1)}, ..., f_{p(n)})$ on the right-hand side. The striped squares have indices in $q(Q \cup \{i\})$ and $q(Q)$, respectively. We must show that the sum under the left part of the bold line is not greater than the sum under the right part. Since

$$\begin{aligned} r_{q(i)} + \sum_{j \in Q} &\leqslant r_{p(i')} + \sum_{j=1}^{i'-1} r_{p(j)} && \text{by the definitions of } i' \text{ and } Q \\ &\leqslant b + \sum_{j=1}^{i'-1} f_{p(j)} && \text{by the assumption} \qquad (1) \\ &= b + \sum_{j \in Q} f_{q(j)} && \text{again by the definition of } Q, \end{aligned}$$

the inequality must hold at least for the striped squares. Using $r_{q(j)} < f_{q(j)}$ for all $1 \leqslant j \leqslant n$ we can add $\sum_{j<i, j \notin Q} r_{q(j)}$ on the left-hand side and $\sum_{j<i, j \notin Q} f_{q(j)}$ on the right-hand side of (1) and thus establish the lemma. ▮

In the subsequent three lemmas p is an **r**-sored and q an **f**-sorted permutation of $\{1, ..., n\}$. For every permutation s of $\{1, ..., n\}$ we have

$$\underset{1 \leqslant i \leqslant n}{\forall} \sum_{j=1}^{i} f_{q(j)} \leqslant \sum_{j=1}^{i} f_{s(j)}. \qquad (2)$$

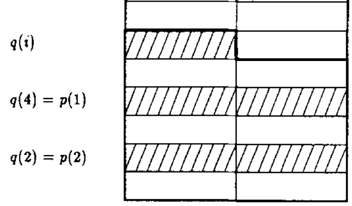

FIG. 3.1. Example of the proof of Lemma 3.1 with $P = \{1, 2\}$ and $Q = \{2, 4\}$.

Lemma 3.1 says that if there is any permutation of $\{1, ..., n\}$ satisyfing the formula in condition (c) of the definition of SAFE (i.e., the condition holds), then the **r**-sorted permutation p does as well. On the other hand, we know from (2) that if p and q as defined above satisfy the formula in condition (c′) of the definition of SAFE′ then the formula must also hold for p and any other permutation. Hence condition (c′) holds in this case. Therefore, in order to prove (c) $\Rightarrow$ (c′) we only have to show that p and q satisfy the formula in (c′) if p does the same for (c).

p will be transformed into q by repeatedly swapping two elements. First $q(n)$ will be shifted to the right, then $q(n-1)$ and so on. For example, $p=(2, 4, 3, 1)$ will be transformed into $q=(1, 2, 3, 4)$ by $(2, 4, 3, 1) \rightarrow (2, 3, 4, 1) \rightarrow (2, 3, 1, 4) \rightarrow (2, 1, 3, 4) \rightarrow (1, 2, 3, 4)$. The resulting sequence of permutations will be called $p=p_0, p_1, ..., p_\tau = q$. It will satisfy

$$\forall_{0 \leqslant \nu \leqslant \tau} \; \forall_{1 \leqslant i \leqslant n} \sum_{j=1}^{i} r_{p(j)} \leqslant b + \sum_{j=1}^{i-1} f_{p_\nu(j)}.$$

LEMMA 3.2. $\forall_{1 \leqslant i \leqslant n} \, r_{p(i)} \leqslant f_{q(i)}$.

Proof. Assume there is an index i with $r_{p(i)} > f_{q(i)}$. From p being **r**-sorted follows

$$|\{j | r_j \leqslant f_{q(i)}\}| < i.$$

Since

$$|\{j | f_j \leqslant f_{q(i)}\}| \geqslant i,$$

there must be an index j with $r_j > f_j$ contradicting condition (a). ∎

LEMMA 3.3. $\forall_{0 \leqslant \nu \leqslant \tau} \, \forall_{1 \leqslant i \leqslant n} \, r_{p(i)} \leqslant f_{p_\nu(i)}$.

Proof. By induction on ν. The lemma holds for $\nu = 0$ by condition (a). Let $i-1$ and i denote the indices swapped in the actual step. By the induction assumption we have

$$f_{p_\nu(i-1)} = f_{p_{\nu-1}(i)} \geqslant r_{p(i)} \geqslant r_{p(i-1)}.$$

Since $r_{p_\nu(i)} = r_{p_{\nu-1}(i-1)}$ is shifted to the right, by definition of the sequence $p_0, p_1, ..., p_\tau$ there exist indices $j \geqslant i$ such that

$$f_{p_\nu(i)} = f_{q(j)} \geqslant r_{p(j)} \geqslant r_{p(i)}.$$

All other indices remain unchanged. ∎

LEMMA 3.4. $\forall_{0 \leqslant \nu \leqslant \tau} \, \forall_{1 \leqslant i \leqslant n} \sum_{j=1}^{i} r_{p(j)} \leqslant b + \sum_{j=1}^{i-1} f_{p_\nu(j)}$.

Proof. By induction on v. The lemma holds for $v=0$ by condition (c). Let $i-1$ and i denote the indices swapped in the actual step. For indices $i' \neq i$ the lemma follows from the induction hypothesis using

$$\sum_{j=1}^{i'-1} f_{p_v(j)} = \sum_{j=1}^{i'-1} f_{p_{v-1}(j)}.$$

For i we deduce

$$\begin{aligned}
\sum_{j=1}^{i} r_{p(j)} &\leqslant r_{p(i)} + b + \sum_{j=1}^{i-2} f_{p_{v-1}(j)} && \text{by the induction hypothesis} \\
&\leqslant f_{p_{v-1}(i)} + b + \sum_{j=1}^{i-2} f_{p_{v-1}(j)} && \text{by Lemma 3.3} \\
&= b + \sum_{j=1}^{i-1} f_{p_v(j)} && \text{by swapping } i-1 \text{ and } i \text{ in the last step.} \quad \blacksquare
\end{aligned}$$

THEOREM 3.5. SAFE = SAFE'.

Proof. Choose $v=\tau$ in Lemma 3.4. ▮

If the vector $\mathbf{r}$ together with the permutations p and q is known, condition (c') can be verified by the following algorithm.

ALGORITHM 1.

```
(*p and q are sorted*)

function test;
begin
   sum = b − r_p(1); i := 1;
   while sum ⩾ 0 and i ⩽ n
   do sum := sum + f_q(i) − r_p(i+1);
      i := i + 1
   od;
      test := sum ⩾ 0
end
```

The time complexity of Algorithm 1 is linear in the number n of clients. If p and q are not constant or incrementally computed, as may occur in some applications, then, in addition, time linear in $n \log n$ is required for sorting.

If Algorithm 1 were based on Theorem 2.4 (instead of Theorem 3.5), in line 7 $f_{q(i)}$ would be replaced by $f_{p(i)}$. If one restricts the algorithm in (Holt, 1972) to problems with only one currency, one gets the same algorithm. But different to (Holt, 1972) we can simplify MIN in the same way as we did for SAFE. Let

$$\mathrm{MIN}' = \Big\{ \mathbf{r} \in \mathbb{N}^n \mid \quad \text{(a)} \quad \forall_{1 \leqslant i \leqslant n} \; r_i \leqslant f_i,$$

$$(\mathrm{b_{min}}) \quad \sum_{i=1}^{n} r_i = \max\left(0, \sum_{i=1}^{n} f_i - g\right),$$

$$(\mathrm{c'_{min}}) \quad \exists p \in \mathrm{PERM}\{1, ..., n\}\; \forall q \in \mathrm{PERM}\{1, ..., n\}$$

$$\forall_{1 \leqslant i \leqslant n} \sum_{j=1}^{i} r_{p(j)} \leqslant \sum_{j=1}^{i-1} f_{q(j)} \Big\}.$$

THEOREM 3.6. MIN = MIN′.

The proof is similar to that of Theorem 3.5.

Note that in condition (c′) of SAFE′ the first two quantifiers can be swapped without changing the set, i.e., SAFE remains unchanged with condition

$$(\mathrm{c}'') :\Leftrightarrow \forall \mathrm{q} \in \mathrm{PERM}\{1, ..., n\}\; \exists p \in \mathrm{PERM}\{1, ..., n\}$$

$$\forall_{1 \leqslant i \leqslant n} \sum_{j=1}^{i} r_{p(j)} \leqslant b + \sum_{j=1}^{i-1} f_{q(j)}$$

instead of (c) and MIN remains unchanged with condition

$$(\mathrm{c''_{min}}) :\Leftrightarrow \forall \mathrm{q} \in \mathrm{PERM}\{1, ..., n\}\; \exists p \in \mathrm{PERM}\{1, ..., n\}$$

$$\forall_{1 \leqslant i \leqslant n} \sum_{j=1}^{i} r_{p(j)} \leqslant \sum_{j=1}^{i-1} f_{q(j)}$$

instead of ($\mathrm{c_{min}}$). In Fig. 2.2 the names SAFE″ and MIN″ are associated with these definitions.

As mentioned in the introduction of this section the following set holds all information that is necessary to describe the set of safe markings:

SORT := {$\mathbf{r} \in \mathbb{N}^n$ | (1) **r** is ordered by increasing numbers;
(2) there is a permutation of **r** in MIN}.

A state **r** can be said to be safe if and only if the sorted permutation $\mathbf{r}'$ of **r** satisfies $\mathbf{r}' \geqslant \mathbf{r}''$ for an $\mathbf{r}'' \in$ SORT. In the example of Section 2 the set of 137 safe or T-continual markings is described by the three elements of SORT = {(0, 1, 9), (0, 2, 8), (0, 3, 7)}.

Algorithm 2 below computes SORT by successively determining all possible values of r_i for every component $1 \leqslant i \leqslant n$. Each time the procedure *recursion* is called in depth i, some values for r_j, $1 \leqslant j \leqslant i$ are given. The varialbe j runs through all numbers for which $(r_1, ..., r_{i-1})$ can be completed to an element $\mathbf{r}$ of SORT. The procedure uses the following variables:

$\text{old} = r_{i-1}$	is the claim of the preceding client (0 for $i = 1$).
$\delta = \sum_{j=i}^{n} r_j$	must give the sum of the claims of the remaining clients j, $i \leqslant j \leqslant n$.
$b = \sum_{j=i}^{n} (f_j - r_j)$	says how much money the banker can give to the remainging clients (as above).
i	counts the depth of the recursion.

By $r_i \geqslant \text{old}$ (lines 11, 13), it is guaranteed that the vector $\mathbf{r}$ is sorted by ascending numbers. By $r_i \leqslant \delta/(n-i+1)$ we can find values $r_j \geqslant r_i$ in the following steps without violating condition $(\mathrm{b}_{\min})$.

By $r_i \leqslant f_i$ condition (a) is fulfilled as well. Like above, $r_i \geqslant f_i - b$ guarantees that we can find $r_j \leqslant f_j$ in the following steps in accordance with condition $(\mathrm{b}_{\min})$.

The last condition in line 12 is the satefy criterion. In Section 2 we have shown that $r_i \leqslant g - b = g - \sum_{j=i}^{n} f_j + \sum_{j=i}^{n} r_j$, which is the same as $\sum_{j=i}^{n} f_j \leqslant g + \sum_{j=i+1}^{n} r_j$, is a reformulation of condition $(\mathrm{c}_{\min})((\mathrm{c}'_{\min})$, respectively.)

ALGORITHM 2.

(* 1*) **begin**
(* 2*)
(* 3*) (*The problem is given by n, g and $\mathbf{f}$. Let $\Delta := \sum_{i=1}^{n} f_i - g$.*)
(* 4*) (*$\mathbf{f}$ is sorted by ascending numbers.*)
(* 5*) (*$\mathbf{r}$ is a global variable for a vector from $\mathbb{N}^n$.*)
(* 6*)
(* 7*) **procedure** *recursion* $(old, \delta, b, i) =$
(* 8*) **if** $i > n$
(* 9*) **then** SORT $:=$ SORT $\cup \{\mathbf{r}\}$
(*10*) **else**

(*11*) **for** j
(*12*) **from max** $(old, f_i - b)$
(*13*) **to min** $(\lfloor \delta/(n-i+1) \rfloor, f_i, g-b)$
(*14*) **do** $r_i := j$;
(*15*) *recursion* $(j, \delta - j, b - f_i + j, i+1)$
(*16*) **od**
(*17*) **fi**;
(*18*)
(*19*) **if** $\Delta < 0$ **then** SORT $:= \{\mathbf{0}\}$
(*20*) **else** SORT $:= \varnothing$,
(*21*) **if** $\forall_{1 \leqslant i \leqslant n} f_i \leqslant g$ **then** *recursion* $(0, \Delta, g, 1)$ **fi**
(*22*) **fi**
(*23*)
(*24*) **end**.

LEMMA 3.7. *The algorithm terminates.*

Proof. The depth of the recusion is limited to $n+1$ by the last parameter. The lower bound of the **for**-loop is at least zero (from the parameter *old*). The upper bound cannot be greater than f_i. Hence the algorithm runs through the loop finitely often. ∎

LEMMA 3.8. *The algorithm computes the set SORT.*

Proof. At first we handle the special cases: If Δ is negative, then ($\mathrm{b}_{\min}$) is equivalent to $\sum_{i=1}^{n} r_i = 0$ and the zero vector is the only element of SORT. If there is an i with $f_i > g$, then SAFE and thus SORT is empty. The algorithm computes the empty set.

Otherwise SORT is computed by the *recursion* procedure. A superscript index $^{(i)}$ means the value of a variable in calling depth i.

($\Rightarrow$) Every vector $\mathbf{r}$ computed by the algorithm belongs to SORT. The parameters *old* assures that $\mathbf{r}$ is sorted. As explained at the beginning of Section 4 it suffices to test for the conditions in the definition of MIN:

(a) $\forall_{1 \leqslant i \leqslant n} r_i \leqslant f_i$ is guaranteed by the second parameter in line 12.

($\mathrm{b}_{\min}$) Both, $r_n \leqslant \delta^{(n)} = \Delta - \sum_{j=1}^{n-1} r_j = \sum_{j=1}^{n} f_j - g - \sum_{j=1}^{n-1} r_j$ and $r_n \geqslant f_n - b^{(n)} = f_n - g + \sum_{j=1}^{n-1} f_j - \sum_{j=1}^{n-1} r_j$ hold certainly. From this follows $\sum_{j=1}^{n} r_j = \sum_{j=1}^{n} f_j - g$ and thus the condition ($\mathrm{b}_{\min}$).

($\mathrm{c}'_{\min}$) From $b^{(i)} = g - \sum_{j=1}^{i-1} f_j + \sum_{j=1}^{i-1} r_j$ and $r_i \leqslant g - b^{(i)} = \sum_{j=1}^{i-1} f_j - \sum_{j=1}^{i-1} r_j$ we can conclude $\sum_{j=1}^{i} r_j \leqslant \sum_{j=1}^{i-1} f_j$ for all $1 \leqslant i \leqslant n$.

($\Leftarrow$) Every vector $\mathbf{r}$ not computed by the algorithm does not belong to SORT. Choose i such that the algorithm finds the first $i-1$ components of $\mathbf{r}$, but then

$$r_i < \min(old, f_i - b)$$

or

$$r_i > \max\left(\frac{\delta}{n-i+1}, f_i, g-b\right) \qquad \text{holds.}$$

(•) If $r_i <$ old then the vector is not sorted.

(•) $r_i < f_i - b^{(i)} = f_i - g + \sum_{j=1}^{i-1} f_j - \sum_{j=1}^{i-1} r_j$ is equivalent to $\sum_{j=1}^{i} r_j < \sum_{j=1}^{i} f_j - g$. Since $r_i \leqslant f_i$ holds for all $1 \leqslant i \leqslant n$, $\sum_{j=1}^{n} r_j$ must be less than $\sum_{j=1}^{n} f_j - g$. This is in contradiction to condition ($\mathrm{b_{min}}$).

(•) Also $r_i > \delta^{(i)}/(n-i+1) = (\Delta - \sum_{j=1}^{i-1} r_j)/(n-i+1)$ and thus $\Delta < (n-i+1)\, r_i + \sum_{j=1}^{i-1} r_j \leqslant \sum_{j=1}^{n} r_j$ contradicts ($\mathrm{b_{min}}$).

(•) $r_i > f_i$ violates the condition (a).

(•) $r_i > g - b^{(i)} = \sum_{j=1}^{i-1} f_j - \sum_{j=1}^{i-1} r_j$ finally is in contradiction to ($\mathrm{c_{min}}$). ∎

LEMMA 3.9. *The algorithm needs time proportional to* $n \cdot |\mathrm{SORT}|$.

Hence the program needs (not considering a constant factor) as much time for computing SORT as a program would need for simply writing down all elements of SORT.

Proof. If the *recursion* procedure is called at all, then $\Delta > 0$ and $\forall_{1 \leqslant i \leqslant n} f_i \leqslant b$ hold. Otherwise there is nothing to prove.

We will show that every call of the procedure at level $i < n$ results in at least one call at level $i+1$. So no level can be called more often than the last one, which writes one element of SORT every time it is called.

We must show that the program runs through the **for**-loop at least once for every call of the procedure in depth $i < n$. Thus we have to show that each of the upper bounds in line 12 is at least as great as the greatest lower bound in line 11.

• $old \leqslant \delta/(n-i+1)$: For $i=1$ this is equivalent to $\Delta = \delta^{(1)} \geqslant 0$. Therefore assume $1 < i \leqslant n$,

$$(n-i+1)\, r_{i-1} = (n-i+1)\, old \leqslant \delta^{(i)} = \delta^{(i-1)} - r_{i-1}$$

$$\Leftrightarrow r_{i-1} \leqslant \frac{\delta^{(i-1)}}{n-i+2}.$$

This inequality is guaranteed by one of the upper bounds in line 12.

- $old \leqslant f_i$: For $i=1$, $0 \leqslant f_i$ by definition. For $i>1$, $old = r_{i-1} \leqslant f_{i-1} \leqslant f_i$.

- $old \leqslant g-b$: For $i=1$, $old=0$ and $g=b^{(1)}$. For $i>1$. $old = r_{i-1} \leqslant g-b^{(i)} = g-b^{(i-1)} + f_{i-1} - r_{i-1}$ is true because $r_{i-1} \leqslant g-b^{(i-1)}$ and $r_{i-1} \leqslant f_{i-1}$.

- $f_i - b \leqslant \delta/(n-i+1)$: One can prove easily by induction that $\delta^{(i)} + b^{(i)} = \sum_{j=i}^{n} f_j$. From this follows $\delta^{(i)} + b^{(i)} \geqslant (n-i+1) f_i$ and consequently $\delta^{(i)} + (n-i+1)\, b^{(i)} \geqslant (n-i+1) f_i$. We obtain the claim by dividing the inequality by $n-i+1$.

- $f_i - b \leqslant f_i$: For $i=1$, the claim $b^{(i)} \geqslant 0$ follows from $g \geqslant 0$. But for $i>1$, $b^{(i)} = b^{(i-1)} - f_{i-1} + r_{i-1} \geqslant 0$ is equivalent to $r_{i-1} \geqslant f_{i-1} - b^{(i-1)}$. This is assured by the second lower bound.

- $f_i - b \leqslant g-b$ will be tested before calling the *recursion* procedure. ▮

4. The Size of SORT

In the preceding chapters we have shown that the set of safe states of the Banker's Problem can be computed directly from the set SORT. In particular, the algorithm mentioned at the end of Section 3 has complexity $n \cdot |\mathrm{SORT}|$. Therefore the size of SORT is of interest.

We deduce some upper bounds for $|\mathrm{SORT}|$ depending on some parameters that describe the size of the problem. In the first part we consider n, the number of clients, and $m = \max_{1 \leqslant i \leqslant n} f_i$, the maximal claim. In the second part we describe the size by n and $s = \sum_{i=1}^{n} f_i$, the sum of the claims.

The capital g is of minor importance since $|\mathrm{SORT}|$ does not grow with g if g is too large, e.g., $|\mathrm{SORT}| = 1$ for $g \geqslant s$. In this section we will try to find a function $f^+(n, m)$, as small as possible such that $|\mathrm{SORT}| \leqslant f^+(n, m)$ holds for all Banker's Problems of size (n, m). We also look for a function $f^-(n, m)$, as great as possible, such that there is a problem satisfying $|\mathrm{SORT}| \geqslant f^-(n, m)$ for every pair (n, m). The same goals are valid for the pair (n, s).

In both parts the deduction will be done in two steps. At first we will derive some functions from the theory of partitions which can serve as f^+ and f^-. The second step will be to find some estimates for these functions.

We assume that the problem is given with sorted claims:

$$\underset{1 \leqslant i \leqslant n}{\forall} \; f_i \leqslant f_{i+1}. \tag{3}$$

Thus id (the identity function) is an f-ordered permutation of $\{1, ..., n\}$. Let $\mathbf{r}$ be an element of SORT. By definition id is also an $\mathbf{r}$-ordered permutation of $\{1, ..., n\}$. Thus by Lemma 3.2 condition (a): $\forall_{1 \leqslant i \leqslant n} \, r_i \leqslant f_i$ holds. Since there is a permutation of $\mathbf{r}$ in MIN and the conditions ($b_{\min}$) and ($c'_{\min}$) do not depend on the order of $\mathbf{r}$, $\mathbf{r}$ itself is an element of MIN and (under assumption (3))

$$\text{SORT} = \{\mathbf{r} \in \text{MIN} \mid \mathbf{r} \text{ is ordered by ascending numbers}\}.$$

If we want to know whether an ordered vector $\mathbf{r}$ belongs to SORT, we can test for membership in MIN directly rather than looking for a suitable permutation. Hereby, when testing condition ($c'_{\min}$), we can assume $p = q = \text{id}$, as explained in the last section.

First we show that from all Banker's Problems of size (n, m) the set SORT becomes maximal for $f_1 = \cdots = f_n = m$.

LEMMA 4.1. *Let A (A') be the Banker's Problem by the parameters n, $\mathbf{f}$ and g (n', $\mathbf{f}'$ and g'). If $n' = n$, $g' = g + 1$, $f'_i = f_i + 1$ for some index i and $f'_j = f_j$ for all $j \neq i$, then*

$$\text{SORT}_A \subseteq \text{SORT}_{A'}.$$

Proof. Comparing the problems A and A', condition ($b_{\min}$) is the same in both cases while the conditions (a) and ($c'_{\min}$) are weaker in the second case. Thus $\text{MIN}_A \subseteq \text{MIN}_{A'}$ holds. From this the lemma follows immediately. ∎

Hence, considering any problem of size (n, m), every f_i can be increased up to m without reducing $|\text{SORT}|$. As long as we look for estimates depending on n and m, we assume $f_1 = \cdots = f_n = m$.

Condition ($c'_{\min}$) of MIN reduces to $r_i = 0$ for $i = 1$. Thus for $i > 1$,

$$\sum_{j=1}^{i} r_j = \sum_{j=2}^{i} r_j \leqslant \sum_{j=2}^{i} f_j = (i-1)\, m = \sum_{j=1}^{i-1} f_j$$

is true in any case.

For $g, n, m \geqslant 0$ we define $L(g, n, m)$ as the set of solutions in $\mathbb{N}^m$ of

$$r_1 + \cdots + r_n = g.$$

satisfying $0 \leqslant r_i \leqslant r_{i+1} \leqslant m$ for every $1 \leqslant i \leqslant n$. Let $P(g, n, m) := |L(g, n, m)|$.

$L^>(g, n, m)$ is defined to contain only the members of $L(g, n, m)$ which satisfy the additional constraint $r_i < r_{i+1}$ for $1 \leqslant i < n$. Finally, $P^>(g, n, m) := |L^>(g, n, m)|$.

Provided $g \leqslant nm$, $|\mathrm{SORT}| = P(nm - g, n-1, m)$ holds for any problem of size (n, m). We do not know to obtain a direct formula for $P(g, n, m)$, but we can describe the function with the help of the next definition and the following two theorems from Andrews (1976): In (Andrews, 1976, Definition 3.1) the Gaussian Polynomial $\left[{n \atop m}\right]$ is defined by

$$\begin{bmatrix} n \\ m \end{bmatrix} := \prod_{i=1}^{m} \frac{1 - x^{n-i+1}}{1 - x^i}$$

for $0 \leqslant m \leqslant n$. If the index range is empty, the product yields 1.

THEOREM 4.2. (Andrews, 1976, Theorem 3.1).

$$\sum_{g=0}^{\infty} P(g, n, m)\, x^g = \prod_{i=1}^{n} \frac{1 - x^{m+i}}{1 - x^i} = \begin{bmatrix} n+m \\ n \end{bmatrix}.$$

In order to compute the value of $P(g, n, m)$, one must calculate the polynomial $\left[{n+m \atop n}\right]$ first. (By Theorem 4.2 it is a polynomial of degree nm.) Now $P(g, n, m)$ is the coefficient of x^g. We will describe the Gaussian polynomial by the following formulas:

THEOREM 4.3 (Andrews, 1976, Theorems 3.2 and 3.10).

(a) $\left[{n \atop 0}\right] = \left[{n \atop n}\right] = 1$;

(b) $\left[{n+m \atop n}\right] = \left[{n+m \atop m}\right]$;

(c) $\left[{n+m \atop n}\right] = \left[{n+m-1 \atop n-1}\right] + x^n \left[{n+m-1 \atop n}\right]$;
$\quad = \left[{n+m-1 \atop n}\right] + x^m \left[{n+m-1 \atop n-1}\right]$;

(d) $\lim_{x \to 1} \left[{n+m \atop n}\right] = [n+m]!/n!\, m! = \binom{n+m}{n}$;

(e) $P(g, n, m) = P(g, m, n)$ (follows from (b));

(f) $P(g, n, m) = P(nm - g, n, m)$;

(g) For $1 \leqslant g \leqslant \lfloor nm/2 \rfloor$: $P(g-1, n, m) \leqslant P(g, n, m)$.

From (a) and (c) we obtain a recursive formula for $P(g, n, m)$:

$$P(0, n, 0) = P(0, 0, m) = 1;$$

$$P(g, n, 0) = P(g, 0, m) = 0 \text{ for } g > 0;$$

$$\begin{aligned} P(g, n, m) &= P(g, n-1, m) + P(g-n, n, m-1) \\ &= P(g, n, m-1) + P(g-m, n-1, m) \end{aligned}$$

$$\text{assuming} \quad P(g, n, m) = 0 \text{ for } g < 0.$$

By (f) and (g),

$$P\left(\left\lfloor \frac{nm}{2} \right\rfloor, n, m\right) = P\left(\left\lceil \frac{nm}{2} \right\rceil, n, m\right)$$

is the maximum of all $P(g, n, m)$ for constant n and m. This leads immediately to

THEOREM 4.4. *For Banker's Problems of size (n, m) with $n, m \geqslant 2$, $P(\lfloor nm/2 \rfloor, n, m)$ is the smallest upper bound for $|\mathrm{SORT}|$.*

Now we have to estimate $P(\lfloor nm/2 \rfloor, n, m)$.

LEMMA 4.5. *For $n, m \geqslant 2$ we have*

$$P\left(\left\lfloor \frac{nm}{2} \right\rfloor, n, m\right) \geqslant \frac{1}{nm} \binom{n+m}{n}.$$

Proof. From Theorem 4.2 and (d) of Theorem 4.3 we obtain

$$\sum_{g=0}^{nm} P(g, n, m) = \lim_{x \to 1} \sum_{g=0}^{\infty} P(g, n, m)\, x^g = \lim_{x \to 1} \begin{bmatrix} n+m \\ n \end{bmatrix} = \binom{n+m}{n}. \tag{4}$$

From the remark following Theorem 4.3 we conclude that $P(\lfloor nm/2 \rfloor, n, m)$ is the maximum of $P(g, n, m)$ for $1 \leqslant g \leqslant nm-1$. Further, it is easy to check that $P(\lfloor nm/2 \rfloor, n, m) \geqslant 2 = P(0, n, m) + P(nm, n, m)$ holds for $n, m \geqslant 2$. Thus $P(\lfloor nm/2 \rfloor, n, m)$ cannot be smaller than the average of these nm terms. Now the lemma follows from (4). ∎

LEMMA 4.6. *For $n \geqslant 1$,*

$$P\left(\left\lfloor \frac{nm}{2} \right\rfloor, n, m\right) \leqslant \frac{1}{m+1} \binom{n+m}{n}.$$

Proof. Let $n \geqslant 1$.

(a) $P(g, n, m) = P^{>}(g + \frac{1}{2} n(n-1), n, m+n-1)$ since we can (similar to the proof of Theorem 4.1 in Andrews, 1976) define a bijection between both sets by $f\colon \mathbf{r} \mapsto (r_1 + 0, r_2 + 1, ..., r_n + n - 1)$.

(b) $P^{>}(g, n, m) \leqslant (1/n) \sum_{i=0}^{\infty} P^{>}(i, n-1, m)$. For each element $\mathbf{r}$ of $L^{>}(g; n, m)$ we can compute n vectors $\mathbf{r}'_i$ from the set $L := \bigcup_{j=0}^{\infty} L^{>}(j, n-1, m)$ by simply omitting the component i of $\mathbf{r}$. Since all r_i, $1 \leqslant i \leqslant n$ are different, we also get n different vectors. We associate them with $\mathbf{r}$. Additionally, no element $\mathbf{r}'$ of L is associated with more than one vector from $L^{>}(g, n, m)$. The inverse image of $\mathbf{r}'$ (if there is one) can be computed for $\mathbf{r}'$ by inserting $g - \sum_{i=1}^{n-1} r'_i$ at the proper place into $\mathbf{r}$.

(c) $\sum_{i=0}^{\infty} P^{>}(i, n-1, m+n-1) = \sum_{i=0}^{\infty} P(i, n-1, m+1)$. We apply the inversion of the function f defined in (a), $f^{-1}(\mathbf{r})$ is equal to $(r_1 - 0, r_2 - 1, ..., r_{n-1} - n + 2)$. Connecting the transformations defined above, we obtain

$$\begin{aligned} P(g, n, m) &\leqslant \frac{1}{n} \sum_{i=0}^{\infty} P(i, n-1, m) = \frac{1}{n} \binom{n+m}{n-1} \\ &= \frac{(n+m)!}{n!(m+1)!} = \frac{1}{m+1} \binom{n+m}{n}. \quad \blacksquare \end{aligned}$$

We summarize the first part of this section in

THEOREM 4.7. *Considering the maximal value of* |SORT| *for all Banker's Problems size* (n, m), *we have the following estimate for* $n \geqslant 3$ *and* $m \geqslant 2$:

$$\frac{1}{(n-1)m} \binom{n+m-1}{n-1} \leqslant |\text{SORT}| \leqslant \frac{1}{m+1} \binom{n+m-1}{n-1}.$$

In the second part we consider estimates depending on n and s. We again derive our formulas from results known from the theory of partitions.

For $g, n \geqslant 0$ we define $L(g, n)$ as the set of solutions of the diophantine equation

$$r_1 + \cdots + r_n = g$$

with $r_1 \leqslant \cdots \leqslant r_n$. $L^+(g, n)$ is defined to be the set of elements of $L(g, n)$ additionally satisfying $\forall_{1 \leqslant i \leqslant n}\, r_i \geqslant 1$. Like before, we define $P(g, n) := |L(g, n)|$ and $P^+(g, n) := |L^+(g, n)|$.

THEOREM 4.8. (a) (Andrews, 1976, Theorem 4.3 (Theorem of Erdös-Lehner)). *As* $g \to \infty$,

$$P^+(g, n) \sim \frac{1}{n!} \binom{g-1}{n-1} \qquad \textit{provided that } n = o(\sqrt[3]{g}).$$

(b) $$\frac{1}{n!} \binom{g-1}{n-1} \leqslant P^+(g, n) \leqslant \frac{1}{n!} \binom{g + \frac{1}{2} n(n-1) - 1}{n-1}.$$

The inequalities from (b) are shown in (Andrews, 1976) in the proof of Theorem 4.3. We do not need part (a) for our proofs, but just mention it to point out that the formula gives a good estimate for $P^+(g, n)$ only if g is not too small compared to n^3.

COROLLARY 4.9.

$$\frac{1}{n!}\binom{g+n-1}{n-1} \leqslant P(g, n) \leqslant \frac{1}{n!}\binom{g+\frac{1}{2}n(n+1)-1}{n-1}.$$

Proof. We define a bijective function between $L(g, n)$ and $L^+(g+n, n)$ by $f(r_1, ..., r_n) := (r_1+1, ..., r_n+1)$. Thus $P(g, n) = P^+(g+n, n)$. ∎

As in the last section, Δ represents the expression $\sum_{i=1}^{n} f_i - g$. We will show that we can find Banker's Problems for any pair (Δ, n) with $n \geqslant 2$, such that $|\text{SORT}| = P(\Delta, n-1)$. Lemma 4.11 will show this fact for $\Delta \not\equiv -1 \pmod{n-1}$, but it will fail for the other case. This gap will be closed by Lemma 4.12. Our final lower bound depending on n and s will then follow from an estimation of $P(\Delta, n-1)$.

LEMMA 4.10. *Let Δ and n be nonnegative integers and $\mathbf{r}$ be the (unique) element of $L(\Delta, n)$ satisfying*

$$\left\lfloor \frac{\Delta}{n} \right\rfloor = r_1 \leqslant r_2 \leqslant \cdots \leqslant r_n = \left\lceil \frac{\Delta}{n} \right\rceil.$$

Then for all $\mathbf{r}' \in L(\Delta, n)$ and all $1 \leqslant i \leqslant n$ we have

$$\sum_{j=1}^{i} r'_j \leqslant \sum_{j=1}^{i} r_j.$$

Proof. If $\forall_{1 \leqslant j \leqslant i}\, r'_j \leqslant r_j$, then nothing remains to be proved. Otherwise we choose j such that $1 \leqslant j \leqslant i$ and $r'_j > r_j$ hold. Then we have $r'_i \geqslant r'_j \geqslant r_j + 1 \geqslant \lceil \Delta/n \rceil$. Thus

$$\sum_{j=i+1}^{n} r'_j \geqslant (n-i)\left\lceil \frac{\Delta}{n} \right\rceil \geqslant \sum_{j=i+1}^{n} r_j.$$

Subtracting this inequality from $\sum_{j=1}^{n} r'_j = \Delta = \sum_{j=1}^{n} r_j$ establishes the claim. ∎

LEMMA 4.11. *For a problem with $n \geqslant 2$ clients, $\Delta \geqslant n-1$, $\Delta \not\equiv -1 \pmod{n-1}$,*

$$\mathbf{f} = \left(\left\lfloor \frac{\Delta}{n-1} \right\rfloor, \left\lfloor \frac{\Delta}{n-1} \right\rfloor, \left\lfloor \frac{\Delta}{n-2} \right\rfloor, \left\lfloor \frac{\Delta}{n-3} \right\rfloor, ..., \left\lfloor \frac{\Delta}{2} \right\rfloor, \Delta\right)$$

and

$$g = \sum_{i=1}^{n} f_i - \Delta$$

we have

$$|\text{SORT}| = P(\Delta, n-1).$$

Proof. Let $L := \{(0, s_1, ..., s_{n-1}) | \mathbf{s} \in K(\Delta, n-1)\}$. We will show that SORT $= L$. Since $\sum_{i=1}^{n} r_i = \Delta$ by condition ($\mathrm{b}_{\min}$) and $r_1 = 0$ by condition ($\mathrm{c}'_{\min}$), SORT $\subseteq L$ is easily shown.

Now consider the opposite direction: Let $\mathbf{r} \in L$ and i an arbitrary index satisfying $1 \leqslant i \leqslant n$. From $\Delta = \sum_{j=1}^{n} r_j \geqslant \sum_{j=i}^{n} r_j \geqslant (n-i+1) r_i$ we know that $r_i \leqslant \lfloor \Delta/(n-i+1) \rfloor \leqslant f_i$ holds. Thus condition (a) from the definition of MIN is proved. ($\mathrm{b}_{\min}$) follows immediately from the definition of L.

The largest part of this proof will handle condition ($\mathrm{c}'_{\min}$). By Lemma 4.10 we may assume that

$$\mathbf{r} = \left(0, \left\lfloor \frac{\Delta}{n-1} \right\rfloor, ..., \left\lceil \frac{\Delta}{n-1} \right\rceil \right).$$

For this case we will show that $\forall_{2 \leqslant i \leqslant n}\, r_i \leqslant f_{i-1}$ holds. Thus condition ($\mathrm{c}'_{\min}$) follows from $r_1 = 0$.

$r_2 = \lfloor \Delta/(n-1) \rfloor = f_1$ is true. Now assume that there is an index i with $3 \leqslant i \leqslant n$ and $r_i > f_{i-1} = \lfloor \Delta/(n-i+2) \rfloor \geqslant \lfloor \Delta/(n-1) \rfloor$. Since on the other hand $r_i \leqslant \lceil \Delta/(n-1) \rceil \leqslant \lceil \Delta/(n-i+2) \rceil$, we obtain

$$r_i - 1 = r_2 = \left\lfloor \frac{\Delta}{n-i+2} \right\rfloor = \left\lfloor \frac{\Delta}{n-1} \right\rfloor \qquad \text{and} \qquad r_i = \left\lceil \frac{\Delta}{n-i+2} \right\rceil = \left\lceil \frac{\Delta}{n-1} \right\rceil. \tag{5}$$

Dividing Δ into $n-1$ or $n-i+2$ parts leads to

$$\Delta = (n-1)\, x_1 + q_1 = (n-i+2)\, x_2 + q_2$$

with $0 \leqslant q_1 < n-1$ and $0 \leqslant q_2 < n-i+2$. Applying (5) gives $x_1 = x_2 = r_2$ and $0 \leqslant q_1 \leqslant q_2 < n-i+2$. But from $r_i > r_2$ follows that all r_j, $i \leqslant j \leqslant n$ must be greater than r_2 as well and

$$\Delta = \sum_{j=2}^{n} r_j \geqslant (n-1)\, r_2 + n - i + 1.$$

Therefore q_1 is at least as great as $n-i+1 \geqslant q_2$. We obtain $q_1 = q_2 = n-i+1$ and $i = 3$. Finally, $q_1 = n-2$ gives $\Delta \equiv -1 \pmod{n-1}$, which was excluded in the assumption of the lemma. ∎

LEMMA 4.12. *For a problem with* $n \geqslant 2$ *clients,* $\Delta \geqslant n-1$, $\Delta \equiv -1 \pmod{n-1}$,

$$\mathbf{f} = \left(\left\lfloor \frac{\Delta}{n-1} \right\rfloor, \left\lfloor \frac{\Delta}{n-1} \right\rfloor, \left\lfloor \frac{\Delta}{n-2}, \left\lfloor \frac{\Delta}{n-3} \right\rfloor, ..., \left\lfloor \frac{\Delta}{2} \right\rfloor, \Delta \right)$$

and

$$g = \sum_{i=1}^{n} f_i - \Delta$$

we have

$$|\mathrm{SORT}| = P(\Delta, n-1).$$

Compared to Lemma 4.12, the value of f_2 changed from $\lfloor \Delta/(n-1) \rfloor$ to $\lceil \Delta/(n-1) \rceil$. Hereby (except for the trivial case $n=2$) f_2 is increased by the one token that was missing in the last proof.

Proof. First we will show that $\lfloor \Delta/(n-1) \rfloor < \lfloor \Delta/(n-2) \rfloor$ and thus $\mathbf{f}$ is still sorted by increasing numbers. Assume $\lfloor \Delta/(n-1) \rfloor = \lfloor \Delta/(n-2) \rfloor =: k$, then $\Delta = (n-1)\,k + q_1 = (n-2)\,k + q_2$ with $q_1 \leqslant q_2 < n-2$ would hold. But since $\Delta \equiv -1 \pmod{n-1}$, q_1 is equal to $n-2$.

Now we can obtain the claim from the proof of the last lemma. The trouble concerning $i=3$ is removed. ▮

We are ready to determine the final lower bound.

THEOREM 4.13. *For each pair (n, s) satisfying $n \geqslant 2$ there is a Banker's Problem with*

$$|\mathrm{SORT}| \geqslant \frac{1}{(n-1)!} \binom{\lfloor s/(\ln n + d_n) \rfloor + n - 2}{n-2}.$$

d_n is a monotonely decreasing sequence of real numbers with $C := \lim_{n \to \infty} d_n \approx 0.577$ (constant of Euler and Mascheroni). For $n \geqslant 4$, d_n is 0.8, $n \geqslant 6 \Leftrightarrow d_n < 0.7$, $n \geqslant 24 \Leftrightarrow d_n < 0.6$.

Proof. Given a pair (n, s), we choose $\Delta := \lfloor s/(\sum_{i=1}^{n-1} 1/i + 1/(n-1)) \rfloor$. Then we have

$$s \geqslant \sum_{i=1}^{n-1} \frac{\Delta}{i} + \frac{\Delta}{n-1} \geqslant \sum_{i=1}^{n-1} \left\lfloor \frac{\Delta}{i} \right\rfloor + \left\lfloor \frac{\Delta}{n-1} \right\rfloor.$$

If $\Delta \geqslant n-1$ and $\Delta \not\equiv -1 \pmod{n-1}$ then there is a vector $\mathbf{f}$ with $\sum_{i=1}^{n} f_i =: s' < s$ that meets the requirements for Lemma 4.11. Hence, there is also a Banker's Problem of size (n, s') with

$$|\mathrm{SORT}| = P(\Delta, n-1) \geqslant \frac{1}{(n-1)!} \binom{\Delta + n - 2}{n-2}.$$

If $\Delta \geqslant n-1$ and $\Delta \equiv -1 \pmod{n-1}$ then $\Delta/(n-1) > \lfloor \Delta/(n-1) \rfloor$ and $s > \sum_{i=1}^{n-1} \lfloor \Delta/i \rfloor + \lfloor \Delta/(n-1) \rfloor$ follow. Since both sides in the last inequality are integers, their difference must be at least 1. By Lemma 4.12 also in this case there is a constant $s' \leqslant s$ and a Banker's Problem of size (n, s') satisfying the same estimate for $|\mathrm{SORT}|$.

In Lemma 4.1 we have shown that in both cases there must be a Banker's Problem of size (n, s) with $|\mathrm{SORT}| \geqslant (1/(n-1)!)\binom{\Delta+n-2}{n-2}$ as well. If $\Delta < n-1$ then $(1/(n-1)!)\binom{\Delta+n-2}{n-2} \leqslant 1$ holds. For every size (n, s) we can find a Banker's Problem with $\mathrm{SORT} = \{\mathbf{0}\}$ be setting $g = s = \sum_{i=1}^{n} f_i$.

The estimate for $\sum_{i=1}^{n-1} 1/i + 1/(n-1)$, namely $\ln n + d_n$, can be obtained, e.g., from (Forster, 1978, Volume I, Problem 20.2). ∎

Considering the upper bound, we first observe that $|\mathrm{SORT}| \leqslant P(s-g, n-1, g)$. We have $s - g = \Delta = \sum_{i=1}^{n} f_i - g$ units of money claims divided into $n-1$ clients $(r_1 = 0)$. Additionally, we may assume that no one of the r_i's is greater than g since otherwise the corresponding f_i would be also greater than g and the Banker's Problem would have no solution at all. Thus

$$\begin{aligned}
\max_{0 \leqslant g \leqslant s} P(s-g, n-1, g) &\leqslant \max_{0 \leqslant g \leqslant s} (\min(P(s-g; g), P(s-g, n-1))) \\
&= \max\left(\max_{0 \leqslant g \leqslant n} P(s-g, g), \max_{n \leqslant g \leqslant s} P(s-g, n-1)\right) \\
&= \max\left(\max_{0 \leqslant g < n} P(s-g, g), P(s-n, n-1)\right)
\end{aligned}$$

is an upper bound for $|\mathrm{SORT}|$. Evaluating this expression for some examples gives fairly good estimates of the real maximal size of SORT. Unfortunately, the formula of Corollary 4.9 gives very bad results if n is great compared to $\sqrt[3]{g}$. The estimate for $P(s-g, g)$ is monotonely ascending with g. Thus we have to be satisfied with an estimate for $P(\Delta, n-1)$.

THEOREM 4.14. *For a Banker's Problem of size (n, s) we have*

$$|\mathrm{SORT}| \leqslant \frac{1}{(n-1)!} \binom{\lfloor (n-1/n)\, s \rfloor + \frac{1}{2} n(n-1) - 1}{n-2}.$$

Proof. We have already shown in the proof of Lemma 4.11 that $|\mathrm{SORT}| \leqslant P(\Delta, n-1)$. Again, we may assume that g is not smaller than $f_1 \leqslant \lceil s/n \rceil$, since otherwise the Banker's Problem had no solution at all. Thus $\Delta = s - g \leqslant s - \lceil s/n \rceil = \lfloor ((n-1)/s) \rfloor$ must hold as well. Now the claim follows from Corollary 4.9. ∎

5. Different Types of Resources

Usually, in operating systems different types of resources, like line printers, tape drivers, or main storage segments are needed. In the Banker's

Problem this corresponds to different currencies of money which cannot be exchanged.

Since the corresponding P/T-net would be very complex, we prefer to model this problem by some type of high level net, namely by a colored Petri-net (Jensen, 1979). This net has three places, BANK, CLAIM, and CREDIT, and two transitions, GRANT and LOAN. It is represented in Fig. 5.1.

In an actual state the banker has b_j units of money of currency j, where $j \in C := \{1, ..., c\}$. Hene the place BANK contains a vector $\mathbf{b} = \{b_1, ..., b_c\}$. (Formally places contain multisets. But we will not distinguish between them and their natural representations as vectors.) A marking in CREDIT is given by a (n, c)-*matrix* $K = (k_{ij})$, where $k_{ij} \in \mathbb{N}$ denotes the actual credit of customer i in currency j. Again, n is the number of customers. In the same way r_{ij} is the remaining claim of customer i in currency j. The corresponding marking is a (n, c)-matrix $\mathbf{R}$ in the place CLAIM. The initial marking M_0 is given by a vector $\mathbf{g} = (g_1, ..., g_c)$ in BANK, a matrix $\mathbf{F} = (f_{ij})$ in CLAIM and the zero matrix of size (n, c) in CREDIT. Transition GRANT has $n \cdot c$ modes of firing, one for each customer and each currency, whereas LOAN has only n modes, since all the money in all currencies must be returned to the banker at once. Since now the formal description of he colored net is straightforward, it is omitted (see Hauschildt, 1985). We refer to this net as the "international" Banker's Problem, in contrast to the "national" version considered in the preceding sections. Similar to the national case, there are two types of invariant equations:

I_{0j}. $\sum_{i=1}^{n} k_{ij} + b_j = g_j$ for each $j \in C$.

I_{ij}. $k_{ij} + r_{ij} = f_{ij}$ for each $i \in \{1, ..., n\}$, $j \in C$.

The following results are similar to those obtained in the "national" case. Therefore, we just mention the representations of the sets IREACH,

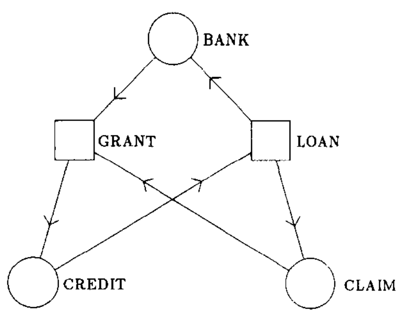

FIGURE 5.1.

ISAFE, and IMIN, but omit the corresponding proofs (again see Hauschildt, 1985).

The vectors $\mathbf{g}$, $\mathbf{b}$, $\mathbf{f}_i$, and $\mathbf{r}_i$ in the folowing definitions are elements of $\mathbb{N}^c$. Operations on them are executed componentwise. Given the conditions

$$\begin{aligned}
&(\mathrm{ia}) \Leftrightarrow && \mathbf{r}_i \leqslant \mathbf{f}_i;\\
&(\mathrm{ib}): \Leftrightarrow && \textstyle\sum_{i=1}^{n} \mathbf{r}_i \geqslant \sum_{i=1}^{n} \mathbf{f}_i - \mathbf{g};\\
&(\mathrm{ib}_{\min}): \Leftrightarrow && \textstyle\sum_{i=1}^{n} \mathbf{r}_i = \max(0, \sum_{i=1}^{n} \mathbf{f}_i - \mathbf{g});\\
&(\mathrm{ic}): \Leftrightarrow && \exists p \in \mathrm{PERM}\{1, ..., n\}\ \forall_{1 \leqslant i \leqslant n} \textstyle\sum_{i'=1}^{i} \mathbf{r}_{p(i')} \leqslant \mathbf{b} + \sum_{i'=1}^{i-1} \mathbf{f}_{p(i')}\\
& && \quad \text{with } \mathbf{b} = \mathbf{g} - \textstyle\sum_{i=1}^{n} \mathbf{f}_i + \sum_{i=1}^{n} \mathbf{r}_i; \text{ and}\\
&(\mathrm{ic}_{\min}): \Leftrightarrow && \exists p \in \mathrm{PERM}\{1, ..., n\}\ \forall_{1 \leqslant i \leqslant n} \textstyle\sum_{i'=1}^{i} \mathbf{r}_{p(i')} \leqslant \sum_{i'=1}^{i-1} \mathbf{f}_{p(i')},
\end{aligned}$$

we define

$$\begin{aligned}
\mathrm{IREACH} &:= \{R \in \mathbb{N}^{nc} \mid (\mathrm{ia}) \text{ and } (\mathrm{ib})\};\\
\mathrm{ISAFE} &:= \{R \in \mathbb{N}^{nc} \mid (\mathrm{ia}), (\mathrm{ib}), \text{ and } (\mathrm{ic})\}; \text{ and}\\
\mathrm{IMIN} &:= \{R \in \mathbb{N}^{nc} \mid (\mathrm{ia}), (\mathrm{ib}_{\min}), \text{ and } (\mathrm{ic}_{\min})\}.
\end{aligned}$$

IREACH represents the set of reachable markings and ISAFE describes the set of safe markings in the net. IMIN is the set of minimal elements of ISAFE.

Now the reader expects, for the "international" case, results similar to Section 3. As before we introduce conditions similar to (c′) and (c″):

$$(\mathrm{ic}') :\Leftrightarrow \begin{matrix}\exists p \in \mathrm{PERM}\{1, ..., n\}\\ \forall q \in \mathrm{PERM}\{1, ..., n\}\end{matrix} \ \underset{1 \leqslant i \leqslant n}{\forall} \sum_{i'=1}^{i} \mathbf{r}_{p(i')} \leqslant \mathbf{b} + \sum_{i'=1}^{i-1} \mathbf{f}_{q(i')};$$

$$(\mathrm{ic}'') :\Leftrightarrow \begin{matrix}\forall q \in \mathrm{PERM}\{1, ..., n\}\\ \exists p \in \mathrm{PERM}\{1, ..., n\}\end{matrix} \ \underset{1 \leqslant i \leqslant n}{\forall} \sum_{i'=1}^{i} \mathbf{r}_{p(i')} \leqslant \mathbf{b} + \sum_{i'=1}^{i-1} \mathbf{f}_{q(i')}.$$

$\mathbf{b}$ can be expressed in terms of $\mathbf{g}$, $\mathbf{F}$, and $\mathbf{R}$ by the same formula as in condition (ic). We also define two sets:

$$\begin{aligned}
\mathrm{ISAFE}' &:= \{R \in \mathbb{N}^{nc} \mid (\mathrm{ia}), (\mathrm{ib}), \text{ and } (\mathrm{ic}')\};\\
\mathrm{ISAFE}'' &:= \{R \in \mathbb{N}^{nc} \mid (\mathrm{ia}), (\mathrm{ib}), \text{ and } (\mathrm{ic}'')\}.
\end{aligned}$$

However, the proofs of Section 3 do not apply here, because the vectors $\mathbf{r}_i$ and $\mathbf{f}_i$ cannot be sorted. Moreover, the corresponding results are not true in this case. We still have the inclusions

$$\mathrm{ISAFE}' \subseteq \mathrm{ISAFE} \subseteq \mathrm{ISAFE}''.$$

The first one follows from $(\mathrm{ic}') \Rightarrow (\mathrm{ic})$ and the second one will be shown in the following theorem.

THEOREM 5.1. ISAFE $\subseteq$ ISAFE$''$.

Proof. We must show, that condition (ic$''$) holds for every safe state R. Let p' be a permutation of $\{1, ..., n\}$, satisfying for every $i \in \{1, ..., n\}$,

$$\sum_{i'=1}^{i} \mathbf{r}_{p'(i')} \leqslant \mathbf{b} + \sum_{i'=1}^{i-1} \mathbf{f}_{p'(i')}. \tag{6}$$

For some arbitrary $q \in \mathrm{PERM}\{1, ..., n\}$ we must find a permutation $p \in \mathrm{PERM}\{1, ..., n\}$ satisfying

$$\sum_{i'=1}^{i} \mathbf{r}_{p(i')} \leqslant \mathbf{b} + \sum_{i'=1}^{i-1} \mathbf{f}_{q(i')}. \tag{7}$$

We will construct $p(1)$ to $p(n)$ step by step such that, for every i, Eq. (7) becomes true and $Q_{i-1} := \{q(1), ..., q(i-1)\} \subseteq \{p(1), ..., p(i)\} =: P_i$ holds.

We define $p(1) := p'(1)$. For $i = 1$ both conditions above are obviously true. After constructing $p(1)$ to $p(i)$ we distinguish between two cases, in order to determine $p(i+1)$.

(1). $q(i) \notin \{p(1), ..., p(i)\}$. We choose $p(i+1) := q(i)$. Since (7) holds for i and $\mathbf{r}_{q(i)} \leqslant \mathbf{f}_{q(i)}$, 7 holds also for $i+1$. $P_{i+1} = P_i \cup \{q(i)\} \supseteq Q_{i-1} \cup \{q(i)\} = Q_i$.

(2) $q(i) \in \{p(1), ..., p(i)\}$. Then $P_i = Q_i$ holds. Let j be the smallest index satisfying $p'(j) \notin P_i$. We define $p(i+1) := p'(j)$. From (6) we know, that

$$\sum_{j'=1}^{j} \mathbf{r}_{p'(j')} \leqslant \mathbf{b} + \sum_{j'=1}^{j-1} \mathbf{f}_{p'(j')}.$$

By adding $\mathbf{r}_{p(i')}$ to the left-hand side of the inequality and $\mathbf{f}_{p(i')}$ to the right-hand side, for each $i' \in P_i \backslash \{p'(1), ..., p(j-1)\}$, we can show (7) to hold for $i+1$. Clearly $Q_i = P_i \subseteq P_{i+1}$. ∎

Figure 5.2 illustrates the proof. It shows $r_{p(1)}, ..., r_{p(n)}$ piled up on the left-hand side and $f_{q(1)}, ..., f_{q(n)}$ on the right-hand side. $p(1)$ is constructed during the initial step which can be seen as a special case to case 2. Afterwards we can proceed according to case 1 until we find the element corresponding to $p(1)$ in q (the striped square). Now $P = Q$ (symbolized by the squares under the bold line). As described in case 2 we use the information from inequality (6) to find $p(i+1)$ and so on.

On the other hand, Theorems 5.2 and 5.3 provide examples in which

$$\text{ISAFE} \not\subseteq \text{ISAFE}' \qquad \text{or} \qquad \text{ISAFE}'' \not\subseteq \text{ISAFE}$$

holds.

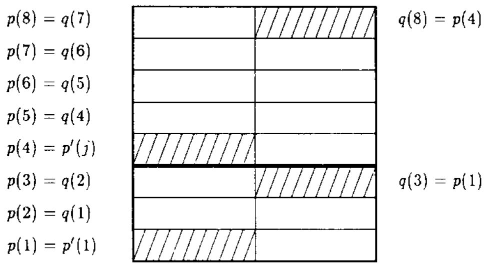

FIG. 5.2. Example of the construction in Theorem 5.1.

THEOREM 5.2. *There is a Banker's Problem satisfing*

$$\mathrm{ISAFE} \not\subseteq \mathrm{ISAFE}'.$$

Proof. The problem is given by n, c, $\mathbf{g}$, and $\mathbf{F}$ with $n=3$, $c=2$, $\mathbf{g}=(1, 1)$, and $\mathbf{F}=((1, 1), (0, 1), (1, 0))$.

$\mathbf{R}=(0, 0), (0, 1), (1, 0))$ belongs to ISAFE, since (ia), (ib), and (ic) hold for $\mathbf{R}$ (with $p=(1, 2, 3)$). But the table of Fig. 5.3 shows a $q \in \mathrm{PERM}\{1, 2, 3\}$ for each $p \in \mathrm{PERM}\{1, 2, 3\}$ contradicting

$$\forall_{i \in \{1, 2, 3\}} \sum_{i'=1}^{i} \mathbf{r}_{p(i')} \leqslant (0, 0) + \sum_{i'=1}^{i-1} \mathbf{f}_{q(i')}.$$

Hence condition (ic$'$) does not hold for $\mathbf{R}$ and thus $\mathbf{R}$ does not belong to ISAFE$'$. ∎

We mention that ISAFE = ISAFE$'$ in the case of $n=2$ customers. The proof is given in (Hauschildt, 1985).

THEOREM 5.3. *There is a Banker's Problem satisfying*

$$\mathrm{ISAFE}'' \not\subseteq \mathrm{ISAFE}.$$

Proof. The Banker's Problem is given by n, c, $\mathbf{g}$, and $\mathbf{F}$ with $n=4$, $c=2$, $\mathbf{g}=(3, 3)$, and $\mathbf{F}=((2, 1), (1, 2), (3, 1), (1, 3))$. $\mathbf{R} := ((2, 1), (1, 2), (3, 0), (0, 3))$ is an element of ISAFE$''$ that does not belong to ISAFE. The con-

p	q
(1,2,3)	(3,1,2)
(1,3,2)	(2,1,3)
(2,1,3)	arbitrary
(2,3,1)	arbitrary
(3,1,2)	arbitrary
(3,2,1)	arbitrary

FIG. 5.3. Example of Theorem 5.2.

q	p
(1,2,3,4), (1,2,4,3)	(2,3,4,1)
(1,3,2,4), (1,3,4,2)	(2,1,3,4)
(1,4,2,3), (1,4,3,2)	(2,1,4,3)
(2,1,3,4), (2,1,4,3)	(1,4,3,2)
(2,3,1,4), (2,3,4,1)	(1,2,3,4)
(2,4,3,1), (2,4,1,3)	(1,2,4,3)
(3,1,2,4), (3,1,4,2)	(2,3,1,4)
(3,2,1,4), (3,2,4,1)	(1,3,2,4)
(3,4,2,1), (3,4,1,2)	(1,3,4,2)

FIG. 5.4. Example of Theorem 5.3.

ditions (ia) and (ib) hold obviously. The table in Fig. 5.4 contains a $p \in \mathrm{PERM}\{1, ..., n\}$ for every $q \in \mathrm{PERM}\{1, ..., n\}$ so that condition (ic″) becomes true. Note that **b** computes as (0, 0, 0, 0) for **R**.

On the other hand, one can easily verify that the marking **R** is not safe. ∎

Again the example is minimal, i.e., for problems with $n \leqslant 3$ clients we have ISAFE = ISAFE″ as it is shown in (Hauschildt, 1985).

RECEIVED January 28, 1986, ACCEPTED July 9, 1986

REFERENCES

ANDREWS, E. (1976), The theory of partitions, *in* "Encyclopedia of Mathematics and Its Applications," Addison–Wesley, Reading, MA.

BRINCH HANSEN, P. (1973), "Operating System Principles," Prentice–Hall, Englewood Cliffs, NJ.

DIJKSTRA, E. W. (1968), Co-operating sequential process; *in* "Programming Languages" (F. Genuys, Ed.), pp. 43–112, Academic Press, London.

FORSTER, O. (1978), "Analysis I, Grundkurs Mathematik," Vieweg, Brunswick.

HAUSCHILDT, D. (1985), "Verklemmungsfreie Betriebsmittelzuteilung mit Hilfe von Residuen," Bericht Fachbereich Informatik, Univesität Hamburg.

HOLT, R. C. (1972), Some Deadlock Properties of Computer Systems, *Comput. Surveys* 4, 179–196.

HOLT, R. C. (1971), "On Deadlocks in Computer Systems," Ph.D. thesis, Cornell Univ., Ithaca, NY; University of Toronto Technical Report CSRG-6.

JENSEN, K. (1979), "Coloured Nets and the Invariant Method," DAIMI PB-104, Aarhus University.

VALK, R., AND JANTZEN, M. (1985), The residue of vector sets with application to decidability problems in Petri nets, *Acta Inform.* **21** (1985), 643–674.